Amendments to Claims:

Listing of Claims:

1. (currently amended): A method in a data processing system, wherein the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code,

wherein the graphical and the textual representations of the source code is generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, and

wherein the graphical representation has portions that correspond to the lines;

initiating an automated process that processes each of the lines; and

while the automated process processes each of the lines,

displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

2. (original): The method of claim 1, further comprising the step of compiling the line before displaying the portion of the graphical representation that corresponds to the line.

3. (original): The method of claim 1, wherein while the automated process processes each of the lines, the method further comprises the step of displaying the line of source code in a visually distinctive manner.

4. (original): The method of claim 1, wherein the graphical representation comprises a class diagram.

5. (original): The method of claim 1, wherein the graphical representation comprises a sequence diagram.

6. (currently amended): A method in a data processing system, wherein the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code,

wherein the graphical and the textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, and

wherein the graphical representation has portions that correspond to the

lines; and

for each of the lines, displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it appears that progression through the code is animated.

7. (original): The method of claim 6, further comprising the step of compiling the line before displaying the portion of the graphical representation that corresponds to the line.

8. (original): The method of claim 6, wherein for each of the lines, the method further comprises the step of displaying the line of source code in a visually distinctive manner.

9. (original): The method of claim 6, wherein the graphical representation comprises a class diagram.

10. (original): The method of claim 6, wherein the graphical representation comprises a sequence diagram.

11. (currently amended): A method in a data processing system, wherein the data processing system comprises source code and the source code comprises a plurality of lines, wherein a graphical and a textual representation of the source code are displayed simultaneously and generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI

class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, the method comprising the steps of:

displaying a graphical representation of the plurality of lines such that at least one of the

lines is not represented in the graphical representation;

initiating an automated process on each of the lines of the source code;

receiving an indication to suspend the automated process when the automated process encounters one of the lines that is represented in the graphical representation; and

while the automated process is being performed on each of the lines of source code, determining

whether the line is represented in the graphical representation; and

when it is determined that the line is represented in the graphical representation, suspending the

automated process.

12. (original): The method of claim 11, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

13. (original): The method of claim 12, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of

displaying the line of source code in a visually distinctive manner.

14. (original): The method of claim 11, further comprising the step of compiling the line before determining whether the line is represented in the graphical representation.

15. (original): The method of claim 11, wherein the graphical representation comprises a class diagram.

16. (original): The method of claim 11, wherein the graphical representation comprises a sequence diagram.

17. (currently amended): A method in a data processing system, wherein the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code, wherein the graphical and the textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, and

initiating an automated process to be performed on each of the lines of the source code; receiving an indication to suspend the automated process when the automated

process encounters a selected one of the lines;

while the automated process is being performed on each of the lines of source code, determining

whether the line is the selected line; and

when it is determined that the line is the selected

line, suspending the automated process.

18. (original): The method of claim 17, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is an mated.

19. (original): The method of claim 18, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the line of source code in a visually distinctive manner.

20. (original): The method of claim 17, further comprising the step of compiling the line before determining whether the line is the selected line.

21. (original): The method of claim 17, wherein the graphical representation comprises a class diagram.

22. (original): The method of claim 17, wherein the graphical representation comprises a sequence diagram.

23. (currently amended): A method in a data processing system, wherein the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code;

wherein the graphical and the textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model;

receiving an indication of a first of the plurality of lines of the source code; selecting a second of the plurality of lines of the source code;

determining whether the second line is the same as the first line; and

when it is determined that the second line is not the same as the first line,

displaying the graphical representation of the second line in a visually distinctive

manner.

24. (original): The method of claim 23, wherein when it is determined that the second line is not the same as the first line, the method further comprises the step of displaying the

second line of the source code in a visually distinctive manner.

25. (original): The method of claim 23, wherein when it is determined that the second line is not the same as the first line, the method further comprises the steps of:

selecting a third of the plurality of lines of the source code;

determining whether the third line is the same as the first line; and

when it is determined that the third line is not the same as the first line,

displaying the graphical representation of the third line in a visually distinctive manner.

26. (original): The method of claim 25, wherein when it is determined that the third line is not the same as the first line, the method further comprises the step of displaying the third line of the source code in a visually distinctive manner.

27. (original): The method of claim 23, wherein the graphical representation comprises a class diagram.

28. (original): The method of claim 23, wherein the graphical representation comprises a sequence diagram.

29. (currently amended): A method in a data processing system, wherein the data processing system comprises source code and the source code comprises a plurality of lines, wherein a graphical and a textual representation of the source code are displayed

simultaneously and generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, the method comprising the steps of:

displaying a graphical representation of the plurality of lines such that at least one of the

lines is not represented in the graphical representation;

initiating an automated process on each of the lines of the source code;

while the automated process is being performed on each of the lines of source code,

compiling the line;

determining whether the compiled line produces an error; and

when it is determined that the compiled line produces the error,

suspending the automated process.

30. (original): The method of claim 29, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

31. (original): The method of claim 30, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of

displaying the line of source code in a visually distinctive manner.

32. (original): The method of claim 29, wherein the graphical representation comprises a class diagram.

33. (original): The method of claim 29, wherein the graphical representation comprises a sequence diagram.

34. (currently amended): A method in a data processing system, wherein the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and textual representation of the source code,

wherein the graphical and textual representations of the source code are

generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model;

selecting one of the plurality of lines of the source code;

compiling the selected line;

determining whether the compiled line produces an error; and

when it is determined that the compiled line does not produce an error, displaying the

graphical representation of the selected line in a visually distinctive manner.

35. (original): The method of claim 34, wherein when it is determined that the compiled line does not produce an error, the method further comprises the step of displaying the selected line of source code in a visually distinctive manner.

36. (original): The method of claim 34, wherein when it is determined that the compiled line does not produce an error, the method further comprises the steps of:

selecting a second of the plurality of lines of the source code;

compiling the second line;

determining whether the compiled second line produces an error; and

when it is determined that the compiled second line does not produce an error, displaying the graphical representation of the second line in a visually distinctive manner.

37. (original): The method of claim 36, wherein when it is determined that the compiled second line does not produce an error, the method further comprises the step of displaying the second line of source code in a visually distinctive manner.

38. (original): The method of claim 34, wherein the graphical representation comprises a class diagram.

39. (original): The method of claim 34, wherein the graphical representation comprises a sequence diagram.

40. (currently amended): A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code,

wherein the graphical and textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, and

wherein the graphical representation has portions that correspond to the lines;

initiating an automated process that processes each of the lines; and

while the automated process processes each of the lines, displaying the portion of the graphical

representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

41. (original): The computer-readable medium of claim 40, wherein the method further comprises the step of compiling the line before displaying the portion of the graphical

representation that corresponds to the line.

42. (original): The computer-readable medium of claim 40, wherein while the automated process processes each of the lines, the method further comprises the step of displaying the line of source code in a visually distinctive manner.

43. (original): The computer-readable medium of claim 40, wherein the graphical representation comprises a class diagram.

44. (original): The computer-readable medium of claim 40, wherein the graphical representation comprises a sequence diagram.

45. (currently amended): A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code,

wherein the graphical and textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, and

wherein the graphical representation has portions that correspond to the lines; and for each of the lines, displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it appears that progression through the code is animated.

46. (original): The computer-readable medium of claim 45, wherein the method further comprises the step of compiling the line before displaying the portion of the graphical representation that corresponds to the line.

47. (original): The computer-readable medium of claim 45, wherein for each of the lines, the method further comprises the step of displaying the line of source code in a visually distinctive manner.

48. (original): The computer-readable medium of claim 45, wherein the graphical representation comprises a class diagram.

49. (original): The computer-readable medium of claim 45, wherein the graphical representation comprises a sequence diagram.

50. (currently amended): A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system comprises source code and the source code comprises a plurality of lines, wherein a graphical and a textual representations of the source code are displayed

simultaneously and generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, the method comprising the steps of:

displaying a graphical representation of the plurality of lines such that at least one

of the lines is not represented in the graphical representation;

initiating an automated process on each of the lines of the source code;

receiving an indication to suspend the automated process when the automated process encounters one of the lines that is represented in the graphical representation; and

while the automated process is being performed on each of the lines of source code,

determining whether the line is represented in the graphical representation; and

when it is determined that the line is represented in the graphical representation,

suspending the automated process.

51. (original): The computer-readable medium of claim 50, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

52. (original): The computer-readable medium of claim 51, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the line of source code in a visually distinctive manner.

53. (original): The computer-readable medium of claim 50, wherein the method further comprises the step of compiling the line before determining whether the line is represented in the graphical representation.

54. (original): The computer-readable medium of claim 50, wherein the graphical representation comprises a class diagram.

55. (original): The computer-readable medium of claim 50, wherein the graphical representation comprises a sequence diagram.

56. (currently amended): A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code;

initiating an automated process to be performed on each of the lines of the source code;

wherein the graphical and textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI

package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta-model;

initiating an automated process to be performed on each of the lines of the source code;

receiving an indication to suspend the automated process when the automated process encounters a selected one of the lines; and

while the automated process is being performed on each of the lines of source code, determining whether the line is the selected line; and

when it is determined that the line is the selected line, suspending the automated process.

57. (original): The computer-readable medium of claim 56, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

58. (original): The computer-readable medium of claim 57, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the line of source code in a visually distinctive manner.

59. (original): The computer-readable medium of claim 56, wherein the method further comprises the step of compiling the line before determining whether the line is the selected line.

60. (original): The computer-readable medium of claim 56, wherein the graphical representation comprises a class diagram.

61. (original): The computer-readable medium of claim 56, wherein the graphical representation comprises a sequence diagram.

62. (currently amended): A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code;

wherein the graphical and textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is storedin a non-repository transient meta model;

receiving an indication of a first of the plurality of lines of the source code; selecting a second of the plurality of lines of the source code; determining whether the second line is the same as the first line; and

when it is determined that the second line is not the same as the first line,

displaying the graphical representation of the second line in a visually distinctive

manner.

63. (original): The computer-readable medium of claim 62, wherein when it is determined that the second line is not the same as the first line, the method further comprises the step of displaying the second line of the source code in a visually distinctive manner.

64. (original): The computer-readable medium of claim 62, wherein when it is determined that the second line is not the same as the first line, the method further comprises the steps of:

selecting a third of the plurality of lines of the source code;

determining whether the third line is the same as the first line; and

when it is determined that the third line is not the same as the first line, displaying the graphical representation of the third line in a visually distinctive manner.

65. (original): The computer-readable medium of claim 64, wherein when it is determined that the third line is not the same as the first line, the method further

comprises the step of displaying the third line of the source code in a visually distinctive manner.

66. (original): The computer-readable medium of claim 62, wherein the graphical representation comprises a class diagram.

67. (original): The computer-readable medium of claim 62, wherein the graphical representation comprises a sequence diagram.

68. (currently amended): A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system comprises source code and the source code comprises a plurality of lines, wherein a graphical and a textual representation of the source code are displayed simultaneously and generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, the method comprising the steps of:

displaying a graphical representation of the plurality of lines such that at least one

of the lines is not represented in the graphical representation; initiating an automated process on each of the lines of the source code;

while the automated process is being performed on each of the lines of source code,

compiling the line;

determining whether the compiled line produces an error; and

when it is determined that the compiled line produces the error,

suspending the automated process.

69. (original): The computer-readable medium of claim 68, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

70. (original): The computer-readable medium of claim 69, wherein while the automated process is being performed on each of the lines of source code, the method further comprises the step of displaying the line of source code in a visually distinctive manner.

71. (original): The computer-readable medium of claim 68, wherein the graphical representation comprises a class diagram.

72. (original): The computer-readable medium of claim 68, wherein the graphical representation comprises a sequence diagram.

73. (currently amended): A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system comprises source code and the source code comprises a plurality of lines, the method comprising the steps of:

displaying simultaneously a graphical and a textual representation of the source code;

wherein the graphical and textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model;

selecting one of the plurality of lines of the source code;

compiling the selected line;

determining whether the compiled line produces an error; and

when it is determined that the compiled line does not produce an error, displaying the graphical representation of the selected line in a visually distinctive manner.

74. (original): The computer-readable medium of claim 73, wherein when it is determined that the compiled line does not produce an error, the method further comprises the step of displaying the selected line of source code in a visually distinctive manner.

75. (original): The computer-readable medium of claim 73, wherein when it is determined that the compiled line does not produce an error, the method further comprises the steps of:

selecting a second of the plurality of lines of the source code; compiling the second line;

determining whether the compiled second line produces an error; and

when it is determined that the compiled second line does not produce an error,

displaying the graphical representation of the second line in a visually distinctive

manner.

76. (original): The computer-readable medium of claim 75, wherein when it is determined that the compiled second line does not produce an error, the method further comprises the step of displaying the second line of source code in a visually distinctive manner.

77. (original): The computer-readable medium of claim 73, wherein the graphical representation comprises a class diagram.

78. (original): The computer-readable medium of claim 73, wherein the graphical representation comprises a sequence diagram.

79. (currently amended): A data processing system comprising:

a secondary storage device further comprising source code wherein the source code comprises a plurality of lines;

a memory device further comprising a program

that displays simultaneously a graphical and a textual representation of the plurality of lines,

wherein the graphical and textual representations of the source code are generated

from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model,

such that at least one of the lines is not represented in the graphical representation,

that initiates an automated process on each of the lines of the source code,

that receives an indication to suspend the automated process when the automated process

encounters one of the lines that is represented in the graphical representation,

that determines whether the line is represented in the graphical representation while the

automated process is being performed on each of the lines of source code, and

that suspends the automated process when it is determined that the line is

represented in the graphical representation, and

a processor for running the program.

80. (original): The data processing system of claim 79, wherein while the automated process is being performed on each of the lines of source code, the program further displays the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

81. (original): The data processing system of claim 80, wherein while the automated process is being performed on each of the lines of source code, the program further displays the line of source code in a visually distinctive manner.

82. (original): The data processing system of claim 79, wherein the program further compiles the line before determining whether the line is represented in the graphical representation.

83. (original): The data processing system of claim 79, wherein the graphical representation comprises a class diagram.

84. (original): The data processing system of claim 79, wherein the graphical representation comprises a sequence diagram.

85. (currently amended): A data processing system comprising:

a secondary storage device further comprising source code wherein the source code comprises a plurality of lines;

a memory device further comprising a program

that displays simultaneously a graphical and a textual representation of the source

code,

wherein the graphical and textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model,

that initiates an automated process to be performed on each of the lines of the source code,

that receives an indication to suspend the automated process when the automated process encounters a selected one of the lines, and

that determines whether the line is the selected line while the automated process is being performed on each of the lines of source code and that suspends the automated process when it is determined that the line is

the selected line; and

a processor for running the program.

86. (original): The data processing system of claim 85, wherein while the automated process is being performed on each of the lines of source code, the program further displays the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

87. (original): The data processing system of claim 86, wherein while the automated process is being performed on each of the lines of source code, the program further displays the line of source code in a visually distinctive manner.

88. (original): The data processing system of claim 85, wherein the program further compiles the line before determining whether the line is the selected line.

89. (original): The data processing system of claim 85, wherein the graphical representation comprises a class diagram.

90. (original): The data processing system of claim 85, wherein the graphical representation comprises a sequence diagram.

91. (currently amended): A data processing system comprising:

a secondary storage device further comprising source code wherein the source code comprises a plurality of lines;

a memory device further comprising a program

that displays simultaneously a graphical and a textual representation of the source code,

wherein the graphical and textual representations of the source code

are generated from a language-neutral representation of the source that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model,

that receives an indication of a first of the plurality of lines of the source code,

that selects a second of the plurality of lines of the source code,

that determines whether the second line is the same as the first line, and

that displays the graphical representation of the second line in a visually distinctive manner when it is determined that the second line is not the same as the first line; and

a processor for running the program.

92. (original): The data processing system of claim 91, wherein when it is determined that the second line is not the same as the first line, the program further displays the second line of the source code in a visually distinctive manner.

93. (original): The data processing system of claim 91, wherein when it is determined that the second line is not the same as the first line, the program further selects a third of the plurality of lines of the source code, determines whether the third line is the same as the first

line, and when it is determined that the third line is not the same as the first line, the program displays the graphical representation of the third line in a visually distinctive manner.

94. (original): The data processing system of claim 93, wherein when it is determined that the third line is not the same as the first line, the program further displays the third line of the source code in a visually distinctive manner.

95. (original): The data processing system of claim 91, wherein the graphical representation comprises a class diagram.

96. (original): The data processing system of claim 91, wherein the graphical representation comprises a sequence diagram.

97. (currently amended): A data processing system comprising:

a secondary storage device further comprising source code wherein the source code comprises a plurality of lines;

a memory device further comprising a program

that displays simultaneously a graphical and a textual representation of the plurality of lines,

wherein the graphical and textual representations of the source code are generated from a language neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI

member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, such that at least one of the lines is not represented in the graphical representation,

that initiates an automated process on each of the lines of the source code,

that while the automated process is being performed on each of the lines

of source code,

the program compiles the line,

determines whether the compiled line produces an error, and

when it is determined that the compiled line produces the error, the

program suspends the automated process; and

a processor for running the program.

98. (original): The data processing system of claim 97, wherein while the automated process is being performed on each of the lines of source code, the program further displays the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated.

99. (original): The data processing system of claim 98, wherein while the automated process is being performed on each of the lines of source code, the program further displays

the line of source code in a visually distinctive manner.

100. (original): The data processing system of claim 97, wherein the graphical representation comprises a class diagram.

101. (original): The data processing system of claim 97, wherein the graphical representation comprises a sequence diagram.

102. (currently amended): A data processing system comprising:

a secondary storage device further comprising source code wherein the source code comprises a plurality of lines;

a memory device further comprising a program

that displays simultaneously a graphical and a textual representation of the source code,

wherein the graphical and textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model,

that selects one of the plurality of lines of the source code,

that compiles the selected line,

that determines whether the compiled line produces an error, and

that displays the graphical representation of the selected line in a visually distinctive manner when it is determined that the compiled line does not produce an error; and

a processor for running the program.

103. (original): The data processing system of claim 102, wherein when it is determined that the compiled line does not produce an error, the program further displays the selected line of source code in a visually distinctive manner.

104. (original): The data processing system of claim 102, wherein when it is determined that the compiled line does not produce an error, the program further selects a second of the plurality of lines of the source code, compiles the second line, determines whether the compiled second line produces an error, and when it is determined that the compiled second line does not produce an error, the program displays the graphical representation of the second line in a visually distinctive manner.

105. (original): The data processing system of claim 104, wherein when it is determined that the compiled second line does not produce an error, the program further displays the second line of source code in a visually distinctive manner.

106. (original): The data processing system of claim 102, wherein the graphical representation comprises a class diagram.

107. (original): The data processing system of claim 102, wherein the graphical representation comprises a sequence diagram.

108. (currently amended): A system having source code wherein the source code comprises a plurality of lines, the system comprising:

means for displaying simultaneously a graphical and a textual representation of the source code,

wherein the graphical and textual representations of the source code are generated from a language-neutral representation of the source code that includes a data structure having a source code interface (SCI) model, an SCI package, an SCI class and an SCI member, wherein the language-neutral representation of the source code is stored in a non-repository transient meta model, and

wherein the graphical representation has portions that correspond to the lines;

means for initiating an automated process that processes each of the lines; and

means for displaying the portion of the graphical representation that corresponds to the line in a visually distinctive manner such that it visually appears that progression of the automated process is animated while the automated process processes each of the lines.

109. (previously presented): The method according to claim 1, wherein the textual representation of the source code is obtained from the source code directly.

110. (previously presented): The method according to claim 6, wherein the textual representation of the source code is obtained from the source code directly.

111. (previously presented): The method according to claim 11, wherein the textual representation of the source code is obtained from the source code directly.

112. (previously presented): The method according to claim 17, wherein the textual representation of the source code is obtained from the source code directly.

113. (previously presented): The method according to claim 23, wherein the textual representation of the source code is obtained from the source code directly.

114. (previously presented): The method according to claim 29, wherein the textual representation of the source code is obtained from the source code directly.

115. (previously presented): The method according to claim 34, wherein the textual representation of the source code is obtained from the source directly.

116. (previously presented): The computer-readable medium according to claim 40, wherein the textual representation of the source code is obtained from the source directly.

117. (previously presented): The computer-readable medium according to claim 45, wherein the textual representation of the source code is obtained from the source directly.

118. (previously presented): The computer-readable medium according to claim 50, wherein the textual representation of the source code is obtained from the source directly.

119. (previously presented): The computer-readable medium according to claim 56, wherein the textual representation of the source code is obtained from the source directly.

120. (previously presented): The computer-readable medium according to claim 62, wherein the textual representation of the source code is obtained from the source directly.

121. (previously presented): The computer-readable medium according to claim 68, wherein the textual representation of the source code is obtained from the source directly.

122. (previously presented): The computer-readable medium according to claim 73, wherein the textual representation of the source code is obtained from the source directly.

123. (previously presented): The computer-readable medium according to claim 79, wherein the textual representation of the source code is obtained from the source directly.

124. (previously presented): The computer-readable medium according to claim 85, wherein the textual representation of the source code is obtained from the source directly.

125. (previously presented): The computer-readable medium according to claim 91, wherein the textual representation of the source code is obtained from the source directly.

126. (previously presented): The computer-readable medium according to claim 97, wherein the textual representation of the source code is obtained from the source directly.

127. (previously presented): The computer-readable medium according to claim 102, wherein the textual representation of the source code is obtained from the source directly.

128. (previously presented): The computer-readable medium according to claim 108, wherein the textual representation of the source code is obtained from the source directly.